AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

LISTING OF CLAIMS:

Claims 1-12 (cancelled).

13. (New) A method for executing a software update of a control unit by flash programming a flash memory of the control unit having multiple segments, via a serial interface, comprising:

establishing requirements to be set for the flash programming;

establishing a flash programming sequence using a finite-state machine which defines states and transitions of the software; and

checking availability, security, and reliability requirements of each state and each transition of the finite-state machine.

14. (New)The method as recited in claim 13, wherein the states and transitions of the software include different operating states including a "starting state," a "normal state," and a "software update," transitions between the operating states, and transition conditions.

15. (New) A method as recited in claim 13, wherein memory arrays of the software of the control unit, which are relevant for flash programming, are divided into programmable and non-programmable memory arrays, and software components to be reprogrammed are correspondingly assigned to the memory arrays.

16. (New) The method as recited in claim 15, wherein the memory arrays of the software are each assigned to a memory of the control unit, one programmable memory array being assigned to at least one segment of the flash memory and one non-programmable memory array being assigned to a ROM of the control unit.

17. (New) The method as recited in claim 13, wherein a boot block, which provides software functionality necessary for executing the flash programming procedure, a program stock and a data stock are stored in segments of the flash memory of the control unit, and a start-up

block, which provides software functionality necessary for executing the flash programming procedure, is stored in a ROM of the control unit.

18. (New) The method as recited in claim 13, wherein security, reliability and availability requirements of the flash programming procedure are specified.

19. (New) The method as recited in claim 14, wherein the "software update" state includes substates, transitions between the substates, and transition conditions.

20. (New) A finite-state machine for executing a software update of a control unit by flash programming, comprising:

an arrangement which defines all substates of the software of the control unit, transitions between the substates, and transition conditions adoptable during execution of the software update; and

an arrangement which specifies permanent, non-erasable storage of a last-valid state or an error-free run state in response to the occurrence of a fault during execution of the software update.

21. (New) The finite-state machine as recited in claim 20, wherein substates for authentication and signature check and substates for the deletion and programming of segments of the flash memory are specified as "abort/error message" and "completion/success message" substates.

22. (New) A memory device storing a computer program, the computer program comprising program code elements via which predefined availability, security, and reliability requirements of each state and each transition of a finite-state machine are automatically checked when the program code elements are executed on a computer or on a computer system, the finite-state machine comprising:

an arrangement which defines all substates of the software of the control unit, transitions between the substates, and transition conditions adoptable during execution of the software update; and

an arrangement which specifies permanent, non-erasable storage of a last-valid state or an error-free run state in response to the occurrence of a fault during execution of the software update.

23. (New) A method for executing flash programming of a boot block which provides software functionality necessary for executing the flash programming procedure and is stored in a first segment of a flash memory, the method comprising:

copying an old boot block to be reprogrammed into a free section of a second memory;

activating the old boot block in the second memory and deactivating the old boot block in the flash memory;

temporarily storing a new boot block in a second segment of the flash memory;

programming the new boot block by copying the second segment into the first segment; and

activating the new boot block in the first segment and deactivating the old boot block in the second memory.

24. (New) The method as recited in claim 23, wherein one boot block is always marked in the flash memory as a valid boot block for restarting the flash programming procedure.